THESIS

RECYCLING DECISION SUPPORT SYSTEM: DESIGN AND DEVELOPMENT OF A WEB-BASED DSS

by

Clayton G. Tettelbach

March, 1997

Thesis Advisor: Hemant K. Bhargava

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE March 97	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE RECYCLING DECISION SUPPORT SYSTEM: DESIGN AND DEVELOPMENT OF A WEB-BASED DSS	5. FUNDING NUMBERS
6. AUTHOR(S) Clayton G. Tettelbach	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

The explosive growth of the World Wide Web creates new opportunities for the development and deployment of Decision Support Systems. No longer restricted by machine-specific limitations, Web-based Decision Support Systems (DSS) provide global access to widely diversified and geographically dispersed users through sharing of data, models, algorithms, and modeling environments. This thesis examines the design and development processes involved in the creation of a Web-based DSS.

The Recycling Decision Support System utilizes a rapid prototype and refinement process to create a Web-based system focusing on supporting ordinary people and industrial users in making good decisions for recycling and disposal of household and industrial waste. Through abstraction of details from the specific Web-based DSS design, a generalized framework for supporting decision-making via the WWW is built which supports functionality in education, queries, and analysis of complex problems.

An important aspect of this research is the development of a new architecture which conforms to the complexities specific to Web-based Decision Support Systems. Prompted by the additional interactions required for WWW connectivity, this architecture incorporates agents for negotiating transactions between the functional components of a standard DSS.

14. SUBJECT TERMS Decision Support System, Internetworking, Design and Programming			15. NUMBER OF PAGES 67
			16. PRICE CODE
17. SECURITY CLASSIFI-CATION OF REPORT Unclassified	18. SECURITY CLASSIFI-CATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICA-TION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

RECYCLING DECISION SUPPORT SYSTEM: DESIGN AND DEVELOPMENT OF A WEB-BASED DSS

Clayton G. Tettelbach
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1984

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
March 1997

Author: [illegible]
Clayton G. Tettelbach

Approved by: [illegible]
Hemant K. Bhargava, Thesis Advisor

[illegible]
Suresh Sridhar, Second Reader

[illegible]
Reuben T. Harris, Chairman, Department of Systems Management

ABSTRACT

The explosive growth of the World Wide Web creates new opportunities for the development and deployment of Decision Support Systems. No longer restricted by machine-specific limitations, Web-based Decision Support Systems (DSS) provide global access to widely diversified and geographically dispersed users through sharing of data, models, algorithms, and modeling environments. This thesis examines the design and development processes involved in the creation of a Web-based DSS.

The Recycling Decision Support System utilizes a rapid prototype and refinement process to create a Web-based system focusing on supporting ordinary people and industrial users in making good decisions for recycling and disposal of household and industrial waste. Through abstraction of details from the specific Web-based DSS design, a generalized framework for supporting decision-making via the WWW is built which supports functionality in education, queries, and analysis of complex problems.

An important aspect of this research is the development of a new architecture which conforms to the complexities specific to Web-based Decision Support Systems. Prompted by the additional interactions required for WWW connectivity, this architecture incorporates agents for negotiating transactions between the functional components of a standard DSS.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENT

The author would like to thank Professor Bhargava for his assistance, guidance and dedication in performing this research. Additionally, a thanks goes to Professor Sridhar for his expertise and inputs into this research. A special thanks to the author's wife, Peggy, and children Carlin, Madison, and Keaton. Their love, support, and patience have been instrumental in making this a tremendous educational experience.

I. INTRODUCTION

The purpose of this thesis is to present, and describe the implementation of, a decision support system (DSS) that is accessible for interaction and use over the World Wide Web (WWW [Ref. 1]). Although there is significant theory and practice related to *desktop* decision support systems, development for Web-based use creates new challenges for DSS developers while delivering new functionality to decision-makers. This thesis discusses the design of such systems, specifically focusing on a system built for supporting the efficient recycling and disposal of household or industrial waste. It also presents a general framework and architecture for implementing Web-based decision support systems.

The *Recycling Decision Support System* is meant primarily for use by creators of household or industrial waste, who have available to them a number of recycling or disposal [1] stations, and who wish to optimize the travel costs and payoffs involved in the recycling or disposal of the waste. In addition, it is useful for anyone interested in getting general information related to recycling, as well as those desiring specific facts about available recycling services in their area. Presently, the system contains data concerning recycling in the Monterey Bay area.

The Recycling Decision Support System maintains a database of recycling stations in a region, including information such as the payoff rates, capabilities, and distances from other recycling stations. Users of the system input their location, items to recycle and amounts of each item, and establish their utility function by answering a question regarding their value for time and travel. Given this data, this system utilizes a mathematical model schema and integer programming algorithms to determine the optimum recycling plan, recommending to the user a set of stations,

[1]Technically, recycling and disposal are separate processes with respect to the reuse of material. One term will be utilized in the remainder of the thesis to refer to both processes because they are functionally equivalent to customers of the DSS.

items to deposit at each station, and the route between stations.

While decision support systems such as this exist for various domains, the Recycling DSS is novel in the following ways. While it resides and executes on a single machine (the *server*), it is accessible, via the World Wide Web, to all Web *clients*. Its user interface is standard Web client software such as the Netscape Navigator. The system communicates its outputs and available command options as HTML pages, extended as required with images and other media, and receives user commands and data via HTML forms. Underlying this, on the server, are a variety of computational components, including a database subsystem, an optimization model, modeling language and integer programming solver, and other executable programs for doing related tasks. The server's Common Gateway Interface (CGI) is used to communicate with these computational subsystems.

A. EXPLOSION OF THE WORLD WIDE WEB

The well publicized and expansive growth of the World Wide Web (W3) [Ref. 1] has taken industry, business and technology leaders by storm. Initially developed to be a pool of human knowledge which would allow collaborators in remote sites to share their ideas and all aspects of a common project, it has emerged as an important medium for communicating information of various sorts [Ref. 1, 2].

The Web is based on technology called hypertext, which allows documents to be connected, or linked, to one another [Ref. 3]. In order to navigate the Web, a hypertext reader or browser is required. A browser interprets and displays hypertext documents; it knows how to find and display a document pointed to by a link. Linked documents on the Web are stored on host computers or servers spread throughout the Internet; the Web provides navigation of the Internet by following the links [Ref. 3].

Although the WWW is playing an increased role in electronic commerce, electronic communications, advertisement and entertainment, few researchers have examined the use of the Web for online, remote processing of computational objects such as mathematical models or algorithms used in decision support systems [Ref. 2].

B. APPLICABILITY OF DECISION SUPPORT SYSTEMS

Decision support systems are "interactive computer based systems which help decision makers utilize data and models to solve unstructured problems" [Ref. 4] and make decisions. The critical and distinctive feature of a DSS is the use of mathematical or other models in the decision-making process. Research in the decision sciences, including operations research and decision analysis, has resulted in the development of a variety of scientific problem-solving and model-based methods for many decision problems faced by individuals and organizations [Ref. 5].

Although a number of computer-based decision technologies are now available in the academic, research, and commercial environments, usage of these decision technologies is far below potential [Ref. 5].

Traditional methods for the distribution of decision support software and scientific expertise require each component of the solution package to be acquired, installed and executed on the users' computational platform[Ref. 5]. This expensive proposition results in problems of awareness, accessibility, compatibility, and applicability [Ref. 5].

Potential DSS users are not always aware of the existence of the relevant decision technologies. Even if they are aware of the technology, they may not have access to, or own a copy of, technologies that might benefit them. Additionally, most of these complex technologies require specific hardware and software which many potential users may not possess [Ref. 5]. These problems in concert with the inherent costs incurred in the development of a decision technology has resulted in a major barrier for optimum use of decision support systems.

C. DSS VIA THE WORLD WIDE WEB

The emergence of the WWW has created a global market for many technologies and products, including decision support systems. The problems discussed previously with standard systems are alleviated through utilization of the Web as the communications medium.

The global reach, ease of use, and electronic search capability of the Web has created a virtual "yellow pages" which alleviates the awareness and advertisement problems inherent in standard Decision Support Systems. The accessibility problem is addressed by the protocols utilized by the Internet for data transport which allow communications between heterogenous networks [Ref. 5]. As exemplified by the immense popularity and traffic statistics, the Web has rapidly become a standard communications medium.

The use of the Web alleviates the requirements for specific hardware and software configurations for each individual system. System users require a Transmission Control Protocol/Internet Protocol (TCP/IP) stack, access to the Internet and a browser to gain Web connectivity. This is rapidly becoming the universal standard configuration for computer systems and provides functionality for more than just a single DSS.

In specifying the requirement for a DSS via the Web, it is assumed that it will provide functionality to a geographically disbursed and diversified group of users. This applicability issue justifies incorporating a DSS capability via the WWW.

Efforts in this area have resulted in projects such as DecisionNet [Ref. 2, 6], an environment that offers access to a distributed library of a class of computational objects useful in decision making [Ref. 2]. Based on the premise of the advantages for providing decision support via the Web, the principle objectives of this initiative are to enable providers of decision technologies to offer their decision technologies in an electronic market and to allow users to make use of these technologies [Ref. 2]. DecisionNet provides a connectivity base for these Web-based independent decision technologies. This thesis will detail the process involved in developing one of these decision technologies.

D. RESEARCH CONTRIBUTIONS AND OBJECTIVES

The objective of this research focuses on the determination of process requirements in the development of a Web-based decision support system. Through rapid application development and deployment of a prototype system, the research was intended to test the feasibility and functionality of a Web-based DSS. Based on a recycling problem representative of a standard decision problem, the prototype evolved into a mature, operational DSS capable of contributing to the recycling needs of the entire Monterey Bay area. With very little effort–precisely, the replacement of Monterey-specific data with data correspnding to another region–the system could be modified to work for any other gographic region.

Iterative prototype development and modification contributed to the creation of a generalized framework for Web-based decision support systems. This framework includes functions in the areas of education, queries and analysis. Examination of the system revealed an architectural modification from that of standard decision support systems. Due to additional protocol translation and data transfer requirements of Web-based systems, the new architecture incorporates the use of functional agents to negotiate communications between major components of a decision support system.

E. OUTLINE FOR THE REMAINDER OF THESIS

Chapter II of this thesis describes the recycling decision problem, purpose of models, and formulation of a mathematical model for solving this problem. Chapter III presents a framework, delineated by functional category, for supporting a Web-based DSS. Chapter IV describes the general architecture of a Web-based DSS and the inherent differences with a standard DSS. Chapter V presents the software agents introduced in the architecture and applied in the Recycling DSS. Finally, Chapter VI will summarize the process, advantages and limitations in the development and application of a Web-based decision support system.

II. MODEL FORMULATION FOR THE RECYCLING PROBLEM

A. THE RECYCLING PROBLEM

Recycling and the manufacture of recycled content products are rapidly maturing forces in the global economic picture [Ref. 7]. Despite this fact, of the one-hundred eighty million tons of waste produced each year by the American population, only thirteen percent is presently recycled [Ref. 8].

As a result, more than two thirds of America's landfills have been lost in the last ten years due to fullness [Ref. 8]. Although not currently in a garbage crisis, America will soon be in a crisis if nothing is done to minimize their landfills [Ref. 8]. Recycling has been recognized as a method that can help solve the problem of crowded landfills, limited or scarce resources, and poisonous substances made from incinerators and landfills [Ref. 8].

> Recycling in California is a growth industry that means new jobs, business opportunities and a healthier environment. The state is home to hundreds of certified recycling centers that collect a variety of materials such as aluminum, plastic and glass. Certified processors, who buy from recyclers and prepare recyclable materials for remanufacture, make up another aspect of the industry. As more and more advances are made in the use of recycled material, the number of manufacturers is on the rise as well. State recycling laws mandate that 50 % of waste be reduced and recycled by the year 2000. That means a big upside in industry growth potential, and estimates predict it will lead to the creation of some 45,000 jobs and an addition of as much as $2 billion to the state's economy. [Ref. 8]

The problem of waste disposal and the need for recycling raises questions regarding why a higher percentage of ordinary people and businesses are not recycling more products. One reason is that the public is not completely or efficiently informed about recycling. If the public was made aware of the various recycling stations available and the materials capable of being recycled, the percentage of waste would decrease and the amount of material recycled would increase.

Secondly, the public must be induced to partake in recycling efforts. The first step has already been taken, recycling stations provide cash incentives for participants that redeem certain valued recyclable products. This collection and redemption of recyclable materials has proved profitable for a wide variety of consumers. The second step is to exploit this monetary incentive through public awareness. Advertising is one alternative, albeit expensive. Awareness achieved through a website not only provides this general functionality, it can provide enhanced explicit value to each customer through interactive interfaces and user input.

The issue raised for the Recycling Decision Support System involves the optimum use of vast amounts of data customized for each specific user instance. The desired information on each recycling station involves the capability to recycle various materials, payoff rates for each material and the location of each station. The underlying assumption is that customers (users of the DSS with materials to be recycled) desire to maximize the profit for recycling all of their materials. The problem specifies that given a customers location, a particular list of materials to recycle and the amounts of each material, which recycling stations should be visited and in what order to maximize the profit for recycling the materials?

B. ROLE OF MODELS

The model is an integral part of any traditional DSS which supports individuals making semistructured decisions. Mathematical models are used for the structured parts, leaving the decisionmaker to exercise judgment in handling the unstructured parts [Ref. 9]. Thus, "the purpose of modeling is not just to get an answer, but also to develop sharper insights into, and understanding of, the problem itself, by examining various facets of it, and by exploring alternative ways of looking at the problem." [Ref. 10]

In the recycling problem, the model was used as a framework for analysis. By defining the parameters, variables, and constraints involved into algebraic equations, a conceptual understanding of the problem was created. In this sense, the model should be a learning device which helps guide the decision maker to examine different viewpoints of the problem.

> A modeling language is designed to express the modeler's form in a way that can serve as direct input to a computer system. Then the translation to the algorithm's form can be performed entirely by computer, without the intermediate stage of computer programming. Modeling languages can help to make mathematical programming more economical and reliable; they are particularly advantageous for development of new models and for documentation of models that are subject to change [Ref. 11].

The Recycling DSS incorporated the use of the AMPL modeling environment, a computational platform for instantiating models with data sets, for selecting and executing solver software, and for formatting and displaying results. The AMPL modeling language is a comprehensive and powerful algebraic modeling language for linear and nonlinear optimization problems, in discrete or continuous variables [Ref. 11].

C. MODEL FORMULATION

1. Notation

- Sets

  $\mathcal{I}$ (ITEMS; index i): Set of Items that need to be disposed or recycled

  $\mathcal{J}$ (STATIONS; indices j and k): Set of Stations that accept items, where j_0 is the location of the waste (i.e. curbside pickup station)

  $\mathcal{J}' = \mathcal{J} - \{j_0\}$

- Exogenous Variables

  $disp_{i,j}$: Total revenue received by customer in disposal of available amount (a_i) of item i at station j; $disp_{i,j_0}$ corresponds to "curbside pickup" for item i

  $trvl_{j,k}$: Travel "cost" between stations j and k; this may be estimated as $b \cdot dist_{j,k}$, where b is the unit travel cost and $dist_{j,k}$ is the Euclidean distance between j and k

  n: number of stations $= |J|$

- Decision Variables: The decision problem is to decide a) the specific assignment of items to stations (and thereby which stations to visit), and b) what route to take in traveling to these stations.

  $x_{i,j}$ (binary integer): 1, if recycling item i at station j

  $y_{j,k}$ (binary integer): 1, if traveling from station j to station k

  u_j: dummy variable used to state subtour elimination constraints

2. Objective

The objective is to find values for the decision variables so as to *Maximize* TP (TotalPayoff), the difference between disposal revenue and travel cost.

$$TP = \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} disp_{i,j} \cdot x_{i,j} - \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{J}} trvl_{j,k} \cdot y_{j,k} \tag{II.1}$$

3. Constraints

We need to ensure that a) all items are recycled, b) every "non-curbside" station at which at least one item is disposed of is traveled to, c) the tours are valid (include the customer's curbside service) and d) there are no subtours. The following constraints are defined.

$$\sum_{j \in \mathcal{J}} x_{i,j} = 1 \quad \forall\, i \in \mathcal{I} \tag{II.2}$$

$$\sum_{k \in \mathcal{J}} y_{j,k} = \sum_{k \in \mathcal{J}} y_{k,j} \quad \forall\, j \in \mathcal{J} \tag{II.3}$$

$$x_{i,j} \leq \sum_{k \in \mathcal{J}} y_{j,k} \quad \forall\, i \in \mathcal{I}, j \in \mathcal{J}' \tag{II.4}$$

$$y_{j,j} = 0 \quad \forall\, j \in \mathcal{J} \tag{II.5}$$

$$\sum_{k \in \mathcal{J}} y_{jo,k} \geq \sum_{k \in \mathcal{J}} y_{j,k} \quad \forall j \in \mathcal{J} \tag{II.6}$$

$$u_j - u_k + n \cdot y_{j,k} \leq n - 1 \quad \forall\, j \in \mathcal{J}', k \in \mathcal{J}' \tag{II.7}$$

Constraints II.5 and II.6 are in fact made redundant due to Constraint II.7, but they are included above for ease of explanation and to tighten the problem definition. Each of the constraints is explained below.

Eq. II.2 : each item is recycled in *exactly one station*; the objective function would ensure its *only* once, but we tighten the constraint to include this condition as well.

Eq. II.3 : you must travel into a station as many times as you travel out of it; in addition, the objective function ensures that each is at most once.

Eq. II.4 : if you recycle anything at all at station j then you must travel to/from j. This is not required for the "home" station, so if everything is recycled at home, there need not be any travel at all.

Eq. II.5 : an easy way to satisfy the previous constraint is to travel from a station to itself; this constraint prevents that, so does the subtour elimination constraint.

Eq. II.6 : Ensures that, if any travel occurs at all, there is travel to/from home, even if nothing is recycled at home; again, the subtour elimination constraint (along with Eq. II.3 and Eq. II.1) ensures this.

Eq. II.7 : There are many ways to prevent subtours (Eq. II.7); we choose the approach described in Lawler [Ref. 12].

III. FRAMEWORK FOR SUPPORTING DECISION-MAKING VIA THE WORLD WIDE WEB

In creating a Web-based decision technology, developers must not only understand the complexities of the decision being solved, they must be able to create a system which correctly visualizes the results and provides information on all facets of the problem. In this sense, they must facilitate cognitive processing by the users. Web connectivity enables a greater customer/client base than any previous DSS could conceivably provide. Instead of just providing a solution to a single decision, a Web-based DSS is capable of interacting with users to answer a myriad of questions, educating users on related problem topics and providing static and dynamic information regarding the subject matter.

In order to assist in the description of the framework for supporting decision-making via the WWW, the waste disposal and recycling problem will be utilized to represent a standard decision problem. The framework can functionally be divided into the areas of education, queries (customized information) and analysis.

A. EDUCATION

Decision support involves more than determining solutions to problems, it must also provide awareness, information and insight into a problem and the world that it represents. Most DSS literature emphasizes the importantance of a system to support individual cognitive processing capabilities and facilitate learning [Ref. 13]. This is the functional role of education, the process of imparting or acquiring general knowledge of the subject and the process in order to develop a systems perspective on the problem.

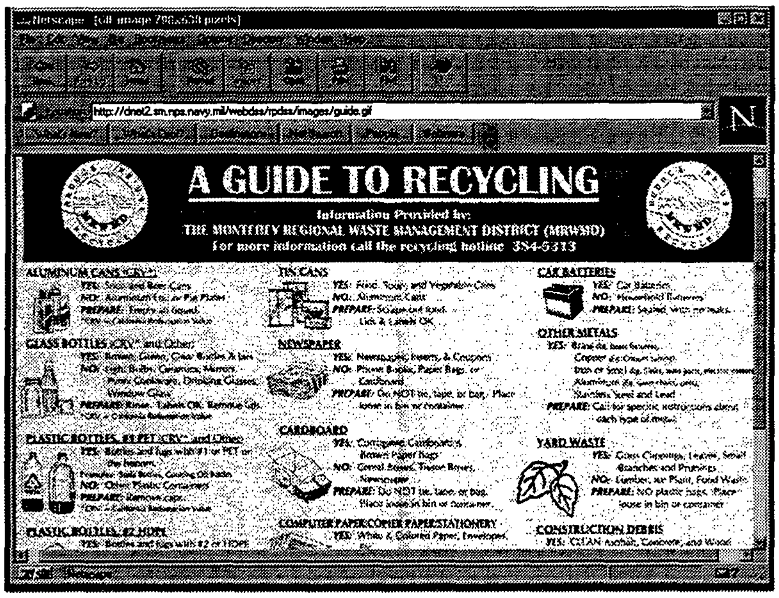
Systems thought emphasizes the need to take a holistic view in order to explain why an object is structured as it is , or how it should be structured. Thus, the system study starts from the outside, identifying the environment in which the object exists and the way it impacts that environment - that is, its role [Ref. 13].

The subject of waste disposal and recycling is one in which most people have some general knowledge of the topic, but a lack of education in the processes involved. The Web has already provided tremendous benefits in this area of education. Although there are volumes of literature on this particular subject and process, it has never been as readily available or accessible to the public as it is today through the WWW.

In the area of Web-based education, information is typically presented to users in the forms of documents, images or graphics. Web documents are written in a standardized language called HTML. HTML, a simple markup language reminiscient of the early word processors, is the "lingua franca" of the Web [Ref. 14]; it identifies the structure of the document and suggests its layout. Servers or hosts store these Web documents and provide them to clients upon request from the clients browser. The display capabilities of the client's Web browser determine the precise appearance of the HTML document on the screen [Ref. 15].

This Web technology provides developers greater flexibility in creating educational resources than any phyical resource can provide. Reproduction of physical information into an HTML formatted document or image is an easy process. Figure 1 displays the "Guide to Recycling" brochure that was scanned into an image file and converted into a source viewable by a Web browser. Easily incorporated into the Recycling DSS, this is but one method of employing printed material as an educational resource.

Whether it be pure text or embedded images, generating static HTML documents is a relatively straightforward and simple process. The creation of HTML editors has transformed this process into one which takes little, if any, knowledge of the HTML language itself. However, while the task is technically straightforward, the challenge is in making good use of the technologies to achieve effective communication.


http://dnet2.sm.nps.navy.mil/webdss/rpdss/images/guide.gif

A GUIDE TO RECYCLING

Information Provided by:
THE MONTEREY REGIONAL WASTE MANAGEMENT DISTRICT (MRWMD)
For more information call the recycling hotline 384-5313

ALUMINUM CANS "CRV"
YES: Soda and Beer Cans
NO: Aluminum Foil or Pie Plates
PREPARE: Empty all liquid.
[illegible]

TIN CANS
YES: Food, Soup, and Vegetable Cans
NO: Aluminum Cans
PREPARE: Scrape out food. Lids & Labels OK

CAR BATTERIES
YES: Car Batteries
NO: Household Batteries
PREPARE: Sealed, with no leaks.

GLASS BOTTLES "CRV" and Other:
YES: Bottles, Glass, Clear Bottles & Jars
NO: Light Bulbs, Ceramics, Mirrors, Pyrex Cookware, Drinking Glasses, Window Glass
PREPARE: Rinse. Labels OK. Remove lids.
[illegible]

NEWSPAPER
YES: Newspaper, Inserts, & Coupons
NO: Phone Books, Paper Bags, or Cardboard
PREPARE: Do NOT tie, tape, or bag. Place loose in bin or container.

OTHER METALS
YES: [illegible]
[illegible]
PREPARE: Call for specific instructions about each type of metal.

PLASTIC BOTTLES, #1 PET "CRV" and Other:
YES: Bottles and Jugs with #1 or PET on the bottom.
[illegible]
NO: Other Plastic Containers
PREPARE: Remove caps.
[illegible]

CARDBOARD
YES: Corrugated Cardboard & Brown Paper Bags
NO: Cereal Boxes, Tissue Boxes, Newspaper
PREPARE: Do NOT tie, tape, or bag. Place loose in bin or container.

YARD WASTE
YES: Grass Clippings, Leaves, Small Branches and Prunings
NO: Lumber, ice Plant, Food Waste
PREPARE: NO plastic bags. Place loose in bin or container.

PLASTIC BOTTLES, #2 HDPE
YES: Bottles and Jugs with #2 or HDPE

COMPUTER PAPER/COPIER PAPER/STATIONERY
YES: White & Colored Paper, Envelopes,

CONSTRUCTION DEBRIS
YES: CLEAN Asphalt, Concrete, and Wood

Figure 1. Guide to Recycling Image

The *Recycling Decision Support System* provides users a process to become educated on issues critical to recycling initiatives. Awareness of environmental issues and government regulations can stimulate interest in the subject. A general guide to materials that are recyclable provides concrete return on investment and value to the Recycling DSS. Figure 1 displays the recycling guide provided in the Education section. Imparting this type of knowledge to users is an essential element of any Web-based DSS.

This system illustrates the value of the Web in providing educational benefits, by informing users on the vast amount of recyclable materials and products. The intention is to encourage users to recycle these materials instead of placing them in waste disposal containers.

Non-textual features such as images can provide additional functionality to educational resources while making them more attractive as well. Visualization is an important aspect in developing attractive and useful web documents. Developers must design documents with a careful mix of text and images which provide insightful enhancements to the contents of the educational information. Image maps may also be incorporated into web documents to facilitate navigation. Through such a map, users can be provided with a graphical overview of any set of information resources; by clicking on different parts of the overview image, they can transparently access any of the information resources (possibly spread out across the Internet) [Ref. 16]. The Web allows highlighted words and pictures in a document to link, or point, to other media. The power of the Web, and what differentiates it from other environments, is that these *hyperlinks* can navigate to other documents in the same directory, anywhere on the same Web server, or anwhere on Web servers or other kinds of servers located anywhere in the world [Ref. 15].

Finally, multimedia in the form of video and audio clips and dynamic images has become a popular feature of many websites. Although primarily utilized for their entertainment value in current sites, they may be incorporated into educational documents to provide additional qualitative information.

Educational resources are typically provided in the form of static documents, but there may be some limited user interactivity incorporated to allow selection of specific features. The increased use of images, graphics and multimedia attracts users and are easily incorporated. Abundant information is available on Web design, HTML authoring and WWW connectivity. The key issues to address in the development of Web pages include excellent, well-maintained content, intuitive site structure and good navigational hyperlinking design [Ref. 17].

B. QUERIES

This is a more interactive section of the system than the Education section, since users will make selections to describe the information they desire. This interactive process applies to both customers of the system and providers of the information.

Interactive Web-based decision support systems more readily spark the inquisitive nature of users than static systems. They also enable users to get information that is customized to their needs. Users should be able to query the DSS for specific data on the subject being supported. This interactive informative process is the basis of the functional area of Queries in a Web-based DSS.

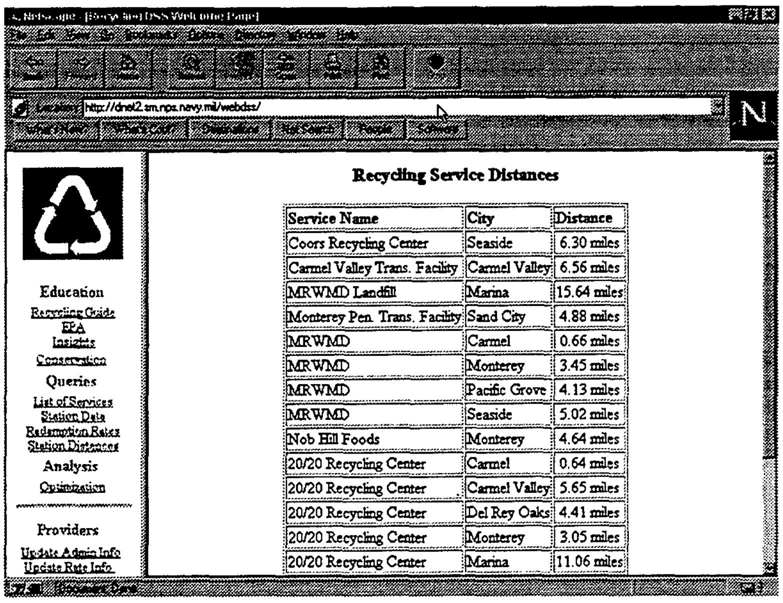
The typical and most efficient method for storing data is through the use of a database. There are other methods available such as file systems and spreadsheets, but neither extends the distinct advantages for compiling, sorting and categorizing information as provided by a relational database. The primary requirement of a database is the ability to communicate with the Web-based information system.

A common method for retrieving this data via the Web is the execution of a Common Gateway Interface (CGI) program. A CGI is a standard for interfacing external applications with information servers. CGI scripts are used to manipulate and retrieve information in the database. These scripts are simply platform-specific executable programs that conform to the platform independent rules of CGI for data exchange [Ref. 18]. A custom CGI program or 'script' is executed in real-time, so that it can output dynamic information [Ref. 16].

Web-based connectivity with databases is becoming a fairly standard feature of interactive websites. Although creating CGI programs and scripts which extract information from databases is by no means a simple endeavor, new tools are continually being developed and commercially produced which simplify the process.

These tools can be utilized to query and retrieve textual or non-textual information. The development difficulty increases signicantly if the information is to be generated in real-time as a function of user-specified data. In designing the Information section of the decision system, the creators and programmers must properly address anticipated client requirements.

The *Recycling DSS* employs interactive features which allow users to obtain a variety of information on the recycling stations in a local area. Specific features include queries designed to provide administrative information such as address and phone numbers of recycling stations as well as the capabilities and payoffs for materials at these stations. Figure 2 displays a unique function of the Queries section.



Recycling Service Distances

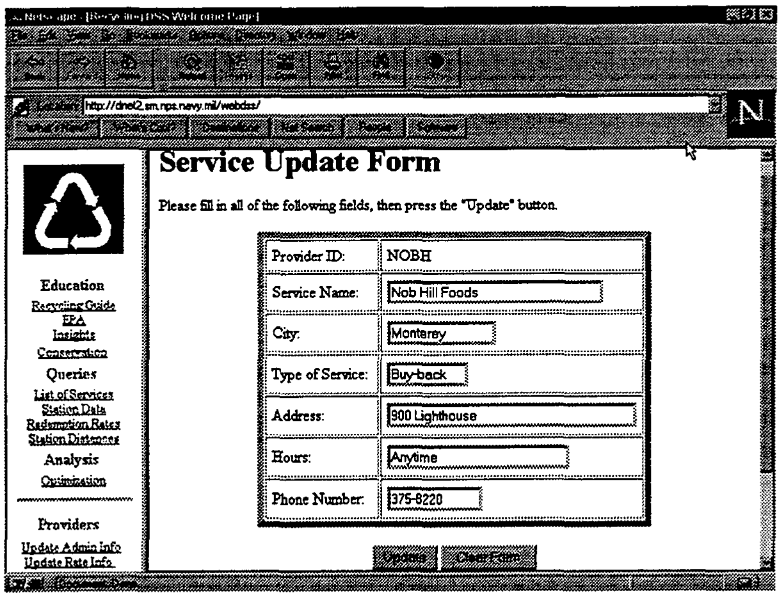
Service Name	City	Distance
Coors Recycling Center	Seaside	6.30 miles
Carmel Valley Trans. Facility	Carmel Valley	6.56 miles
MRWMD Landfill	Marina	15.64 miles
Monterey Pen. Trans. Facility	Sand City	4.88 miles
MRWMD	Carmel	0.66 miles
MRWMD	Monterey	3.45 miles
MRWMD	Pacific Grove	4.13 miles
MRWMD	Seaside	5.02 miles
Nob Hill Foods	Monterey	4.64 miles
20/20 Recycling Center	Carmel	0.64 miles
20/20 Recycling Center	Carmel Valley	5.65 miles
20/20 Recycling Center	Del Rey Oaks	4.41 miles
20/20 Recycling Center	Monterey	3.05 miles
20/20 Recycling Center	Marina	11.06 miles

Figure 2. Recycling Service Distances Query Results

This interactive query is initiated by the user clicking a geographical location on an image map. When the user clicks on a specific position, information on the distances to all of the recycling stations in a local area is presented. This provides users an efficient method of being informed on the closest stations without having to interpret address information.

The interactive design should incorporate features to allow providers to retrieve, update and supply information as well as other functions unique to provider requirements. Again, these functions primarily take the form of CGI scripts, customized for a specific decision support system. Such unique features are incorporated in the *Recycling DSS.*

Providers are allowed to update their administrative information as well as the payoff rates they establish for recycling various materials. This allows providers to remotely update their information universally after a simple login and password verification process. An example of a provider update web page is provided in Figure 3. Providers simply type any modified information into the text areas provided and select update to change the data in the database tables.



http://dnet2.sm.nps.navy.mil/webdss/

Service Update Form

Please fill in all of the following fields, then press the "Update" button.

Provider ID:	NOBH
Service Name:	Nob Hill Foods
City:	Monterey
Type of Service:	Buy-back
Address:	900 Lighthouse
Hours:	Anytime
Phone Number:	375-8220

Figure 3. Recycling Services Administrative Update Form

C. ANALYSIS

The final functional area incorporated into the decision support system is the Analysis section. This is the most sophisticated section of the system which integrates the use of a model to solve a complex decision problem. Any support beyond direct access to raw data requires the application of a model [Ref. 13]. The ability to invoke, run, change, combine, and inspect models is a key capability in DSS and therefore a core service [Ref. 13]. Although acute knowledge of the decision problem is a general knowlege requirement of the developer, rapid application development tools and other technological innovations have made the previous functions user friendly. In the Analysis section, developers will be required to build and incorporate sophisticated algorithms for analysis of unique problems, hence it is the most difficult area of application development.

Integration of the model and a computational application to run the model is the basic function of the analysis function. The developer must properly incorporate interfaces and supporting hypertext to provide the greatest user functionality in the Analysis section.

> Many computational system developers view hypertext only in terms of accessing and managing documents (or smaller units of static information). Such display-oriented behavior characterizes the majority of hypertext systems. They are designed to facilitate authoring, creating relationships, displaying information, and navigating through large information spaces. In computational applications, on the other hand, document management and object display are second in importance. Here, hypertext must augment both interface and analytical activities. This requires that instead of adding computation to a hypertext application, hypertext must be integrated into the design of the computational application. [Ref. 19]

In the Analysis section, the developer must incorporate the collection of the data that defines a specific problem instance, run a computational application of the model with the collected data, and generate an output of the results.

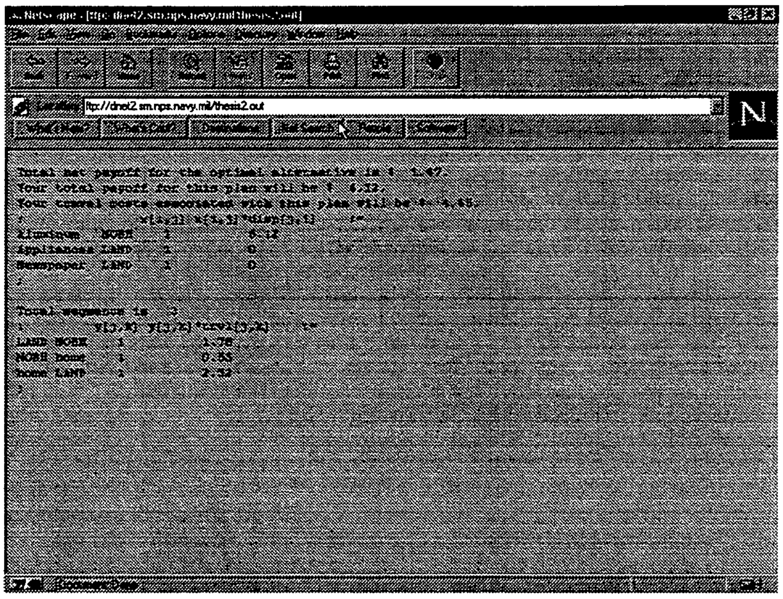
Collection of the data results from a combination of user input and data stored in a database. In the Web-based system, the user inputs are typically obtained through the use of forms. Handling form input is one of the most common uses of CGI scripts today. This is in large part due to the numerous uses for forms. A form is just a group of HTML tags that generate such elements as input fields, list boxes, check boxes, radio buttons, and push buttons [Ref. 20]. Forms provide a useful means of retrieving textual input, but input representations need not be limited to being textual. Although more complex to integrate, representations such as image maps may be utilized which provide interesting functionality to the system. The Recycling Decision Support System, for example, incorporates the use of an image map to retrieve location data on the user.

In the Recycling DSS, users are required to enter information, specific to their problem, such as user location, types of materials to be recycled, and quantities of each material to be recycled. This information is retrieved through the use of the forms and utilized by the CGI scripts to perform queries on the database tables to generate the data supplied to the mathematical program. The complexity arises not only in the generation of the model in the computational application, but in the supply of the data in the proper format to the application. The data must be retrieved from the database as well as manipulated to conform to the requirements of the specific modeling language utilized by the application. The application must then be sent commands to execute with the proper model and data set which adds to the complexity of the problem.

This also leads to the issue of state maintenance within the DSS. HyperText Transfer Protocol (HTTP) is the supporting protocol for transfering information within the WWW. It is also a stateless protocol, one in which there is no record, or 'memory' of a connection from one request for information to the next [Ref. 21]. The advantage of this stateless protocol is that it can run faster because it does not have to maintain extra information. Conversely, more information must be transferred with each connection to report necessary data from prior transactions [Ref. 21]. In order to alleviate this problem in a Web-based DSS, the system must be able to solve the problem using one transaction with the user. The data must be retrieved and passed with each transaction or each session with the system must maintain state through use of a file or record which stores the session data. The Recycling DSS uses a hybrid method. User input is utilized to retrieve data from the database and is transferred with each connection for some basic functions. Once the data is sent to the optimization model, a file is created spefic to the user and this file is utilized for subsequent transactions including the solution output.

The output of the problem must be reformatted in a manner such that the user will be able to correctly interpret the results. This is another complicated evolution for the developer because it involves retracing the steps from hypertext to computational application in the reverse manner. The results of a scientific computational application are typically generated in a fashion capable of interpretation only by a person familiar with the modeling language. Therefore, the developer must generalize the output and construct it in a format meaningful to the users of the system.

In the Recycling DSS, the output of the optimization routine executed by the AMPL modeling language and solver software utilizes extensive algrabraic notation



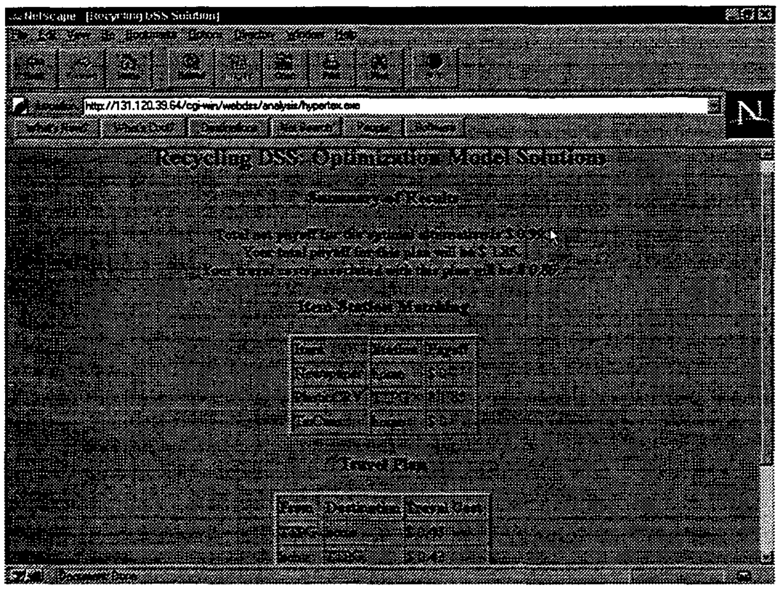
```
Total net payoff for the optimal alternative is $ 1.47.
Your total payoff for this plan will be $ 6.12.
Your travel costs associated with this plan will be $-4.65.
:           x[i,j] x[i,j]*disp[i,j]    :=
Aluminum   NOBH    1          6.12
Appliances LAND    1          0
Newspaper  LAND    1          0
;

Total sequence is   3
:         y[j,k] y[j,k]*trv1[j,k]    :=
LAND NOBH    1          1.78
NOBH home    1          0.55
home LAND    1          2.32
;
```

Figure 4. AMPL Generated Output

as displayed by Figure 4.

In order to make this information useful, a CGI script was created to reformat the output using HTML tags in a form that makes sense to the user. The output clearly delineates the optimized total payoff amount, total travel cost incurred and net payoff value. Additionally, the hypertext converted output provides the user with the optimized route traversed, services visited, and the materials recycled at the services visited as seen in Figure 5. Although not graphically enhanced, the output is provided in a form useful to the users involved.


Netscape - [Recycling DSS Solution]

http://131.120.39.64/cgi-win/webdss/analysis/hypertex.exe

Recycling DSS: Optimization Model Solutions

Summary of Results

[illegible]

Item-Station Matching

Travel Plan

Document: Done

Figure 5. Hypertext Converted Output

IV. ARCHITECTURE FOR A WEB-BASED DSS

From an architectural viewpoint, a Web-based DSS departs in certain ways from a standard desktop DSS. Components of a standard DSS, represent a functional breakdown of the system with the assignment of specific software modules mainly a question of arrangement and resource allocation[Ref. 13]. Direct communications between components is built into the system. Creation of a Web-based DSS, perhaps with the components distributed over the network, requires the transfer and translation of data across several applications and languages. The architecture of Web-based decision support systems, then, must be extended to include *agents* to broker transactions between the standard functional components.

A. STANDARD DSS ARCHITECTURE

Most DSS design literature identifies three major functions or conceptual components necessary for a DSS [Ref. 13]. These components include a database management system (DBMS), a model management system (MMS) and a user interface management system (UIMS) as displayed in Figure 6.

The data management or DBMS component reflects a fundamental aspect of the role of DSS—all levels of decision support are based on access to a set of data

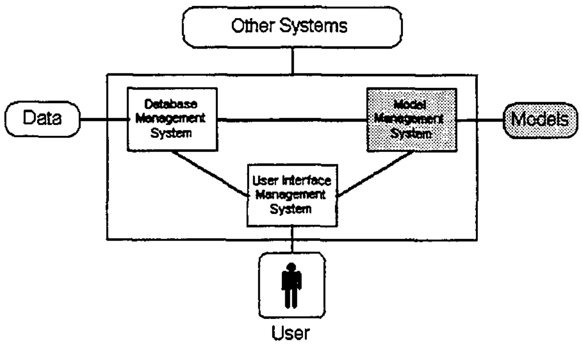


Figure 6. Standard DSS Architecture

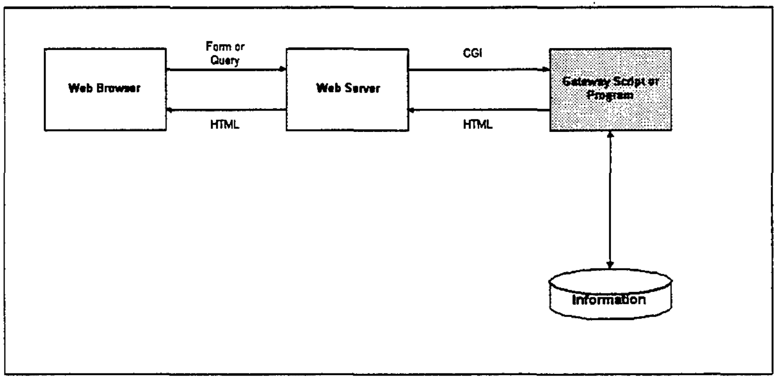


Figure 7. The CGI Model

[Ref. 13]. This component captures and stores data as well as information about the data itself. Specifically, it includes the database, a data directory, a query facility and a staging and extraction function for accessing sources and neighboring systems.

The mechanism for explicit management of models and modeling activity is what distinguishes a DSS from other traditional information processing systems [Ref. 13]. The ideal MMS should provide the ability to capture, store and catalog models, as well as information about the models. It should provide model execution control, model command processing and a database interface for data retrieval and output.

The dialogue between the user and the system establishes the framework in which outputs are presented as well as the context for user inputs[Ref. 13]. Functionally, the UIMS should accept requests, provide responses, present data and models and provide assistance.

B. WEB-BASED DSS AGENTS

In contrast to a standard DSS, communications between the UIMS, MMS, and DBMS in Web-based DSS must flow back and forth over the Internet in a client/server architecture. Each instance of analytical processing by a Web-based DSS requires information to flow from the Web browser to the server, through a gateway script or program, to the information source or application and back again as seen in Figure 7.

Instead of direct communications between functional components, a Web-

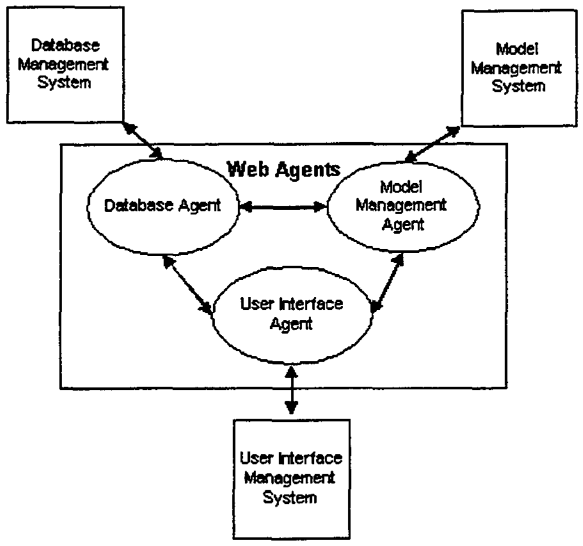


Figure 8. Web-based DSS Architecture

based DSS utilizes a set of agents to negotiate transactions and communications between components. These functional agents are required because of the additional protocols and data translations involved in making Web documents actively interactive and communicable with various components of a DSS.

The increased communication layers require the introduction of Web agents to function as communication, translation and data transfer brokers between standard functional components. These agents include a database agent, a model management agent, a user interface agent and a Web agent as depicted in Figure 8.

1. Database Agent

Functionally independent from a DBMS, a database agent serves the purpose of communicating requests and commands to the DBMS as required by the other functional agents. The database agent interprets requests for specific services of the database, such a an SQL statement, and formats the request in a form understandable by the DBMS. The agent will, in turn, broker the response of the DBMS and communicate the output to other effected agents. In general, the DSS agents, DBMS and MMS may be located on different machines. In this configuration, the DBMS has no direct communications or transactions with the MMS or UIMS. Required capabilities of the database agent are:

- to provide a database and DBMS access mechanism for the DSS,
- query translation to interpret requests for data,
- data translation to interpret query results and formulate results for use by other agents.

In the Recycling DSS, the database agent plays a role, for example, in the retrieval (from the database) of payoff rates associated with a specific service. This information is later used to create a data file for each specific instance of an optimization routine.

The data file's contents are determined in part from user inputs which are retrieved from an HTML form by the user interface agent. This agent interprets the results and negotiates a request for data from the database agent. Based on the user inputs, the database agent formats a specific query for the DBMS. The results of the query are returned to the database agent and through communications with the model management agent, a data file is generated. The structure of the data file is based on the data format requirements of the modeling language, the model and the solver software. In this instance, the database agent negotiated all transactions between the functional components.

2. Model Management Agent

The ability to invoke, run, change, combine, and inspect models is a key capability in any DSS and therefore a core service. Any support beyond direct access to raw data requires the application of a model [Ref. 13]. In a standard DSS, this functionality would be provided solely by the model management system.

A benefit of a Web-based DSS is the capability to execute and solve decision problems from a remote location. In order to do this, the modeling environment must be capable of handling information generated from web document forms and database retrieval mechanisms. This can be a difficult process because modeling languages typically require models and data sets to be in specifically predefined formats for proper execution. Integration of the model, data, and execution program through a Web-based interface is critical. The function of the model management agent is to compile, translate, interpret, format and execute the model and data in the proper instance. The ideal model management agent should provide:

- a model command processor to accept and interpret modeling instructions and to route them to the MMS
- an interpreter to accept database information and reformat the data for processing by the specific model being executed
- an output generator to store model outputs for perusal by system users.

The AMPL modeling environment provides separation of model and data. The model management agent in the Recycling DSS acts as the facilitator in matching model and data files for execution. A model file was created to define the recycling and waste disposal problem using mathematical programming. A data file in prescribed format is generated 'on the fly' for each instance of user interaction. The model management agent insures validity of the data file and compliance with AMPL modeling environment format restrictions. Finally, the model management agent creates a batch file 'on the fly' along with the data file. The batch file includes the commands required to run the program with the appropriate model and data files, call the appropriate solver, and output the results to a file defined and created by the agent.

3. User Interface Agent

The rules of user interface design include techniques for handling interaction. The Web brings together a wide range of disciplines, including cognitive design, desktop publishing layout, graphics design, 3-D virtual world design, and interactivity challenges akin to computer game playing, hypertext theory, library sciences, and classical document management [Ref. 17]. The design and development of the user interface will require the developer to experience or make a simulated environment in which the user feels comfortable, challenged, interested, productive, and entertained.

The dialogue between the user and the system establishes the framework in which outputs are presented as well as the context for user inputs. In a Web-based DSS, the user interface is instantiated by a clients Web browser and the format is defined by the underlying HTML code retrieved from the server for each request. In order to communicate with the other components of a Web-based DSS, a user interface agent is utilized to facilitate interactions. Since the Web was originally designed as a scheme of passive hypertext documents, the user interface agent allows for active interaction between clients of a DSS and the DSS itself. Required functions of the user interface agent include:

- a dialogue-control function to determine the basic semantics of interactions and maintain the interaction context
- a request transformer to provide the necessary translations from users' vocabulary to the system's other agents vocabulary
- an output translator to reformat system generated output into hypertext.

Overall design of the Recycling DSS user interface supports a common theme which is maintained throughout the web pages. The Recycling DSS maintains a design supporting the use of frames to separate user control functions from document presentation. The design enhances the functionality of the system by providing users constant access to DSS functions as long as they maintain navigation within the DSS web site domain.

The forms generated for user input provide the access mechanism necessary to stimulate interaction with the DSS. The user interface agent in the Recycling DSS determines the format of the forms to be displayed to the user as well as retrieval of form data and sends it to the database agent for facilitating queries. On the output end of the system, the user interface agent accepts optimization model generated output and reformats it into a dynamic hypertext document viewable by a Web browser and understandable by system users.

C. USE OF AGENTS: AN EXAMPLE

The use of these functional agents can be seen in many instances of the Recycling DSS. A specific example of agent interaction is displayed through the use of an image map to negotiate client input. This specific function of the DSS allows users to determine the closest recycling stations in the local area by clicking the mouse button on their present location on a geographical image map.

Instead of providing navigation to another web document or web site, the pixel coordinates are parsed out by the user interface agent and sent to the model management agent. The database agent is passed a query request to retrieve pixel coordinate information and passes the data to the model managment agent. The model management agent sends both sets of data and a model request to the MMS for execution. Results of the calculations are passed through the model management agent to the user interface agent which turns the data around into HTML syntax. The HTML formatted data is sent to the server and passed along to the clients browser completing the processing and communications loop.

V. SOFTWARE AGENTS FOR THE RECYCLING DSS

Web browsers can directly access several types of Internet information services, but not all local information sources fit the Web authoring mold. Gateways solve the problem of dynamic information generation and database access by providing an extension mechanism for the Web server. In practice, a gateway is just a script or program invoked by the Web server, that can accept user input through the Web server and can output HTML, a Universal Resource Locator (URL), or some other data back to the user through the Web server [Ref. 15].

Forms provide an efficient manner to collect data from users. The Web server passes form input to a script or program on the system, which processes the data. The HTML part of the form is known as the *front end*, while the script that handles the input is known as the *back end*. Most, but not all, Web servers have two basic methods of handling *back-end* data. They either read a file or communicate with other programs through the *Common Gateway Interface* (CGI). The CGI is the mechanism for communciating between a gateway and a Web server. It's up to the application to figure out what to do with that data, and to return HTML to the server [Ref. 22]. These CGI scripts or programs are the cornerstone applications for a Web-based DSS. The scripts perform as the software agents described in the architecture to allow communication between the system, the server and the client.

A. RECYCLING DSS SOFTWARE AGENT DEVELOPMENT TOOLS

Common Gateway Interface scripts can be written in almost any programming language. CGI in itself is not a programming language, therefore developers must know one in order to write CGI scripts.

Borland's Delphi Client/Server Suite Version 2.0 was utilized as the Rapid

Application Development and database development tool for creating all of the CGI scripts in the Recycling DSS. It is a visual, object-oriented, component-based development environment which utilizes Object Pascal as the underlying programming language for the source code of the executable programs. Database development refers to the fact that Delphi is an ideal tool to design database client applications that communicate with a variety of local or remote databases [Ref. 23]. The addition of CGI, CGI Database and External file execution components [Ref. 24] allows for a seamless interface between users and the system via the WWW [Ref. 18]. These components provided the communications mechanisms of the functional agents.

Utilization of Borland's Paradox for Windows 95 Version 7 as the database for the Recycling DSS provides enhanced functionality to the programming process due to the inherent product connectivity between the database and the development tool.

B. RECYCLING DSS ANALYTICAL FLOW

The Recycling DSS provides users with solutions to specific recycling and disposal problems dependent on variables inputted to the system. These variables include user location, materials and quantities to be recycled, and user utility for travel distance.

User location is obtained through an image map as seen in Figure 9, while other input values are incrementally obtained and passed to subsequent programs through standard dynamically generated forms.

The users location is passed in pixel coordinates as a CGI Query String to a software agent. The agent parses out the data into *X* and *Y coordinates* which are used against a database query of recycling station pixel coordinates to calculate straight line distances incorporating a *Pythagorean theorem* model (Equation V.1).

$$z = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad \text{(V.1)}$$

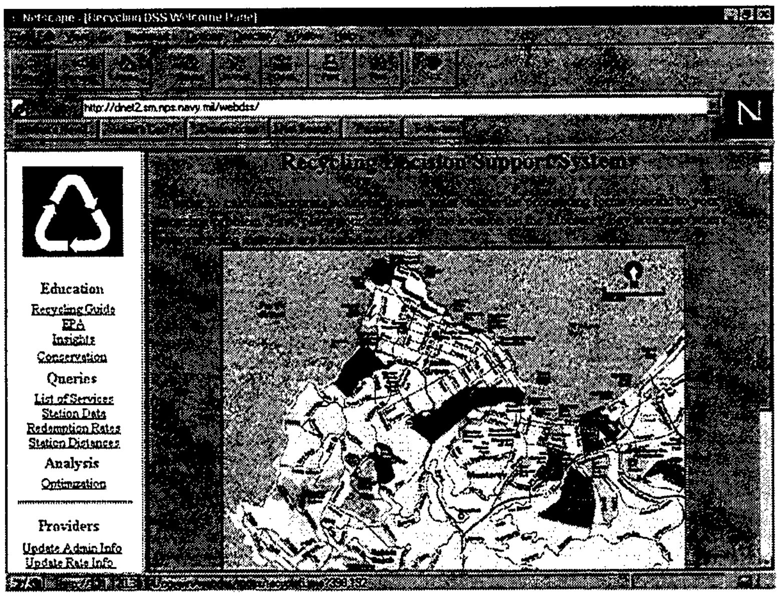


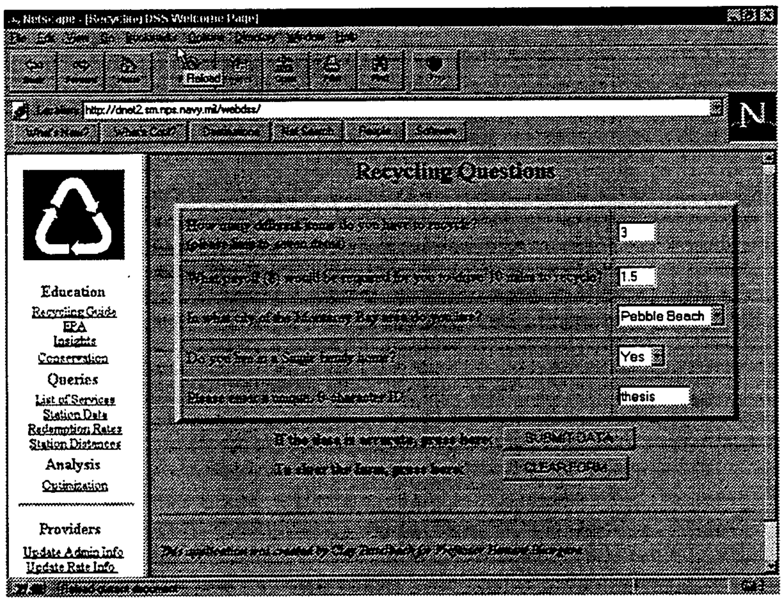
Figure 9. Recycling DSS Image Map

In Equation V.1 let (x_1, y_1) be the user coordinates and (x_2, y_2) be the station coordinates, then z will equal the Euclidean straight line distance in pixels.

The user interface agent creates forms to accept user input for the number of items to recycle, user value for a standard travel distance, as well as other inputs required for data retrieval queries specific to unique recycling station alternatives as displayed in Figure 10.

The users utility for travel is calculated using a straight linear relationship between distance and cost. Assuming zero travel equates to zero cost, a single point is identified on a two-axis graph by the users response to the utility question. The slope (m) calculated in Equation V.2 for the line created by the connection of the zero point (0,0) and the users input (x_1, y_1) becomes the linear relationship necessary for converting travel distances to monetary utility (U) in the model agent using a pixel-to-miles conversion factor (p) as shown in Equation V.3.

$$m = \frac{y_1 - 0}{x_1 - 0} = \frac{y_1}{x_1} \tag{V.2}$$


Netscape - [Recycling DSS Welcome Page]

Location: http://dnet2.sm.nps.navy.mil/webdss/

Recycling Questions

How many different items do you have to recycle? (please keep to seven items)	3
What payoff ($) would be required for you to drive 10 miles to recycle?	1.5
In what city of the Monterey Bay area do you live?	Pebble Beach
Do you live in a Single family home?	Yes
Please enter a unique 8-character ID:	thesis

If the data is accurate, press here: SUBMIT DATA

To clear the form, press here: CLEAR FORM

Education
Recycling Guide
EPA
Insights
Conservation
Queries
List of Services
Station Data
Redemption Rates
Station Distances
Analysis
Optimization
Providers
Update Admin Info
Update Rate Info

This application was created by [illegible]

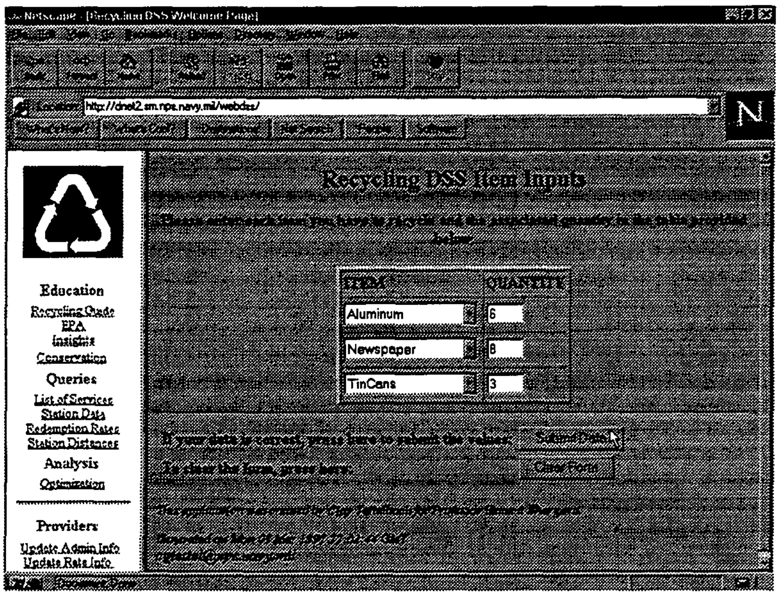
Figure 10. Recycling DSS Questions

$$U = z \cdot m \cdot p \tag{V.3}$$

The inputted number of recycling materials is used by the software agent to generate the form for required user input of types of material and associated quanities as displayed in Figure 11. This input is retrieved by the user interface agent and passed to the database management agent for query formulation and execution.

The data gathered from the user is passed along to each script through the use of *hidden fields.* The resultant of final form for user input is a dynamically created page which provides the user with a list of optional analysis features. These features allow the user to choose the type of solution to calculate and provide as seen in Figure 12. These options include a Lazy Heuristic, Payoff Matrix and Optimization as well as other future additions.

The lazy heuristic utilizes the user data to generate query request by the database agent on the recycling stations which have the capability to recycle the items specified as well as payoff rates and distances for those stations. The results are passed to the model management agent which formats the data. The agent then



Location: http://dnet2.sm.nps.navy.mil/webdss/

Education
Recycling Guide
EPA
Insights
Conservation
Queries
List of Services
Station Data
Redemption Rates
Station Distances
Analysis
Optimization
Providers
Update Admin Info
Update Rate Info

Recycling DSS Item Inputs

Please select each item you have to recycle and the associated quantity in the table provided below.

ITEM	QUANTITY
Aluminum	6
Newspaper	8
TinCans	3

If your data is correct, press here to submit the values: Submit Data

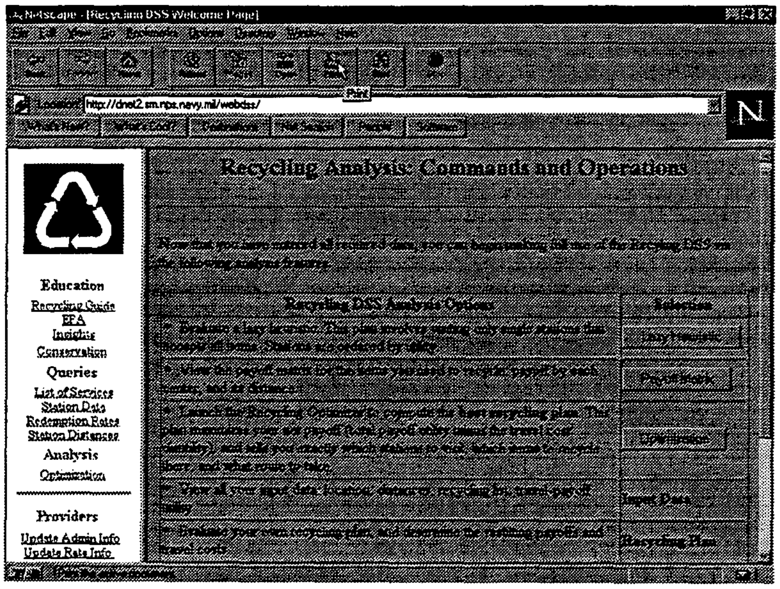
To clear the form, press here: Clear Form

This application was created by [illegible]

[illegible]

Document: Done

Figure 11. Recycling DSS Item Inputs



Location: http://dnet2.sm.nps.navy.mil/webdss/

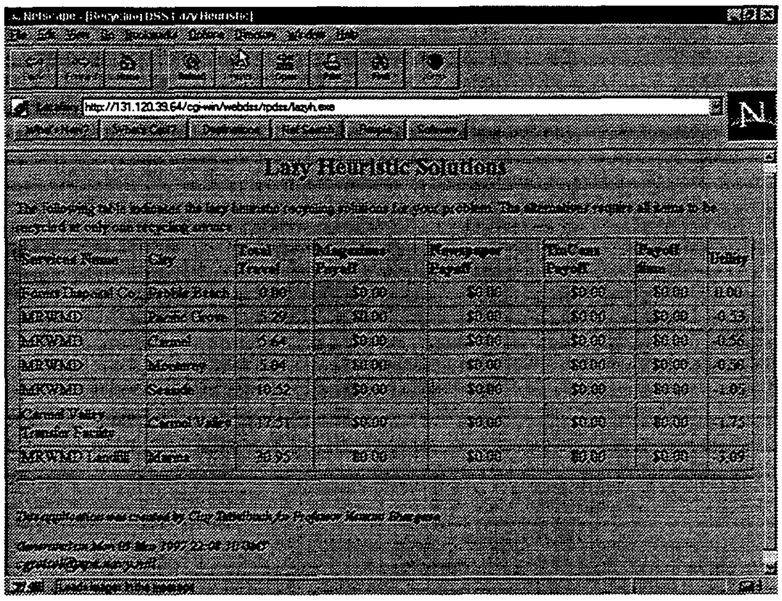
Education
Recycling Guide
EPA
Insights
Conservation
Queries
List of Services
Station Data
Redemption Rates
Station Distances
Analysis
Optimization
Providers
Update Admin Info
Update Rate Info

Recycling Analysis: Commands and Operations

Now that you have entered all required data, you can begin making full use of the Recycling DSS via the following analysis features.

Recycling DSS Analysis Options	Selection
Evaluate a lazy heuristic. This plan involves visiting only single stations that accept all items. Stations are ordered by utility.	Lazy Heuristic
View the payoff matrix for the items you need to recycle: payoff by each station, and its distance.	Payoff Matrix
Launch the Recycling Optimizer to generate the best recycling plan. This plan maximizes your net payoff (total payoff minus the travel cost penalty), and tells you exactly which stations to visit, which items to recycle there, and what route to take.	Optimization
View all your input data: location, distance, recycling list, travel-payoff utility	Input Data
Evaluate your own recycling plan, and determine the resulting payoffs and travel costs	Recycling Plan

Figure 12. Recycling DSS Analysis Options


http://131.120.39.64/cgi-win/webdss/rpdss/lazyh.exe

Lazy Heuristic Solutions

The following table indicates the lazy heuristic recycling solutions for your problem. The alternatives require all items to be recycled at only one recycling service.

Services Name	City	Total Travel	Magazines Payoff	Newspaper Payoff	Tin Cans Payoff	Payoff Sum	Utility
[illegible] Disposal Co.	Pebble Beach	0.00	$0.00	$0.00	$0.00	$0.00	0.00
MRWMD	Pacific Grove	5.29	$0.00	$0.00	$0.00	$0.00	-0.53
MRWMD	Carmel	5.64	$0.00	$0.00	$0.00	$0.00	-0.56
MRWMD	Monterey	5.84	$0.00	$0.00	$0.00	$0.00	-0.58
MRWMD	Seaside	10.52	$0.00	$0.00	$0.00	$0.00	-1.05
Carmel Valley Transfer Facility	Carmel Valley	17.51	$0.00	$0.00	$0.00	$0.00	-1.75
MRWMD Landfill	Marina	30.95	$0.00	$0.00	$0.00	$0.00	-3.09

This application was created by [illegible] for Professor [illegible]

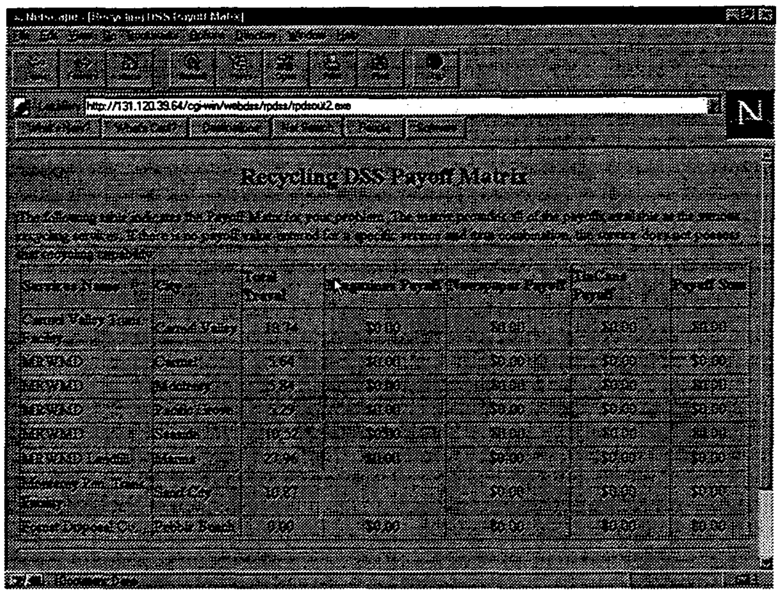
[illegible]

Figure 13. Recycling DSS Lazy Heuristic

passes the data to the appropriate model to perform payoff and utility calculations. The model orders the results by utility value and passes the data back to the user interface agent through the model management agent. The solutions are reformatted in HTML by the user interface agent and displayed on the users browser as seen in Figure 13.

The payoff matrix provides the user with the calculated payoffs at each recycling station for the items specified as displayed in Figure 14. All item-specific payoffs rates and coordinate information for recycling stations are retrieved from the query and passed along to the model management agent. The model management agent accumulates and structures all of the data and instructs the model management system to execute payoff and travel distance calculations with the appropriate model. The payoff and distance data is transfered to the user interface agent for formatting and a payoff matrix is created and returned in HTML as seen in Figure 14.

Selection of optimization returns the best possible solution taking into account payoffs, utility, and possible visitation of more than one recycling station. In addition to steps performed in the payoff matrix program, the optimization program requests



http://131.120.39.64/cgi-win/webdss/rpdss/rpdsout2.exe

Recycling DSS Payoff Matrix

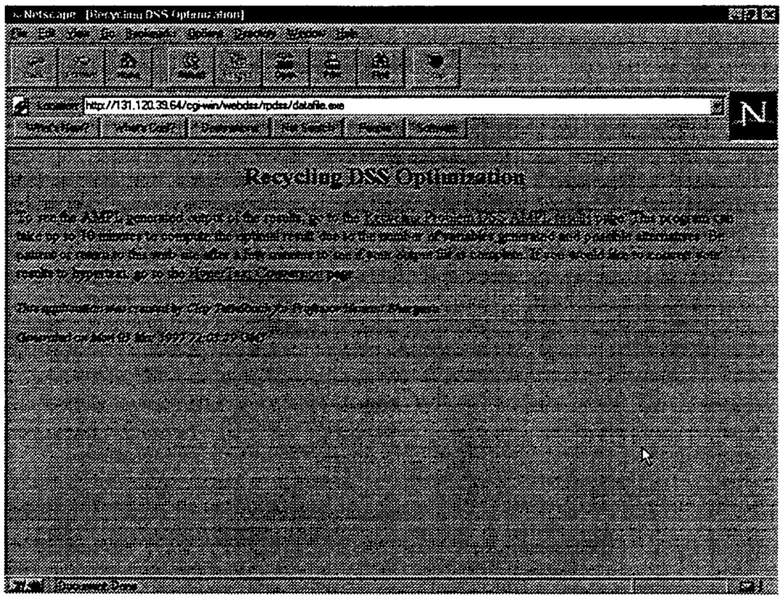
The following table indicates the Payoff Matrix for your problem. [illegible]

Servicer Name	City	Total Travel	[illegible] Payoff	Newspaper Payoff	Tin Cans Payoff	Payoff Sum
Carmel Valley Trans. Facility	Carmel Valley	18.74	$0.00	$0.00	$0.00	$0.00
MRWMD	Carmel	7.64	$0.00	$0.00	$0.00	$0.00
MRWMD	Monterey	5.88	$0.00	$0.00	$0.00	$0.00
MRWMD	Pacific Grove	7.25	$0.00	$0.00	$0.00	$0.00
MRWMD	Seaside	10.52	$0.00	$0.00	$0.00	$0.00
MRWMD Landfill	Marina	22.96	$0.00	$0.00	$0.00	$0.00
Monterey [illegible]	Sand City	10.87		$0.00	$0.00	$0.00
[illegible] Disposal Co.	Pebble Beach	0.00	$0.00	$0.00	$0.00	$0.00

Figure 14. Recycling DSS Payoff Matrix

additional queries from the database management agent to retrieve distance data between recycling stations. The model management agent formats the cumulative query results into a data file required by the modeling environment. The model management agent also generates a batch file with the required execution and display commands required by the specific instance of the model being executed. The agent communicates the appropriate commands to the AMPL modeling environment for execution. The model management system returns an output file solution to the model management agent. The agent passes the location of the file to the user interface agent and provides the user the option of viewing the AMPL generated output or creating a hypertext tagged and formatted version as depicted by Figure 15.

Selection of hypertext conversion executes a parsing program by the user interface agent. This program takes the AMPL generated output file from the model management agent and parses out the required data for the overlying text and algebraic symbology. The agent embeds HTML tags and reformats the output. The optmized solution is returned to the client in HTML with all of the data pertinent to the the specified problem as seen in Figure 16.



http://131.120.39.64/cgi-win/webdss/rpdss/datafile.exe

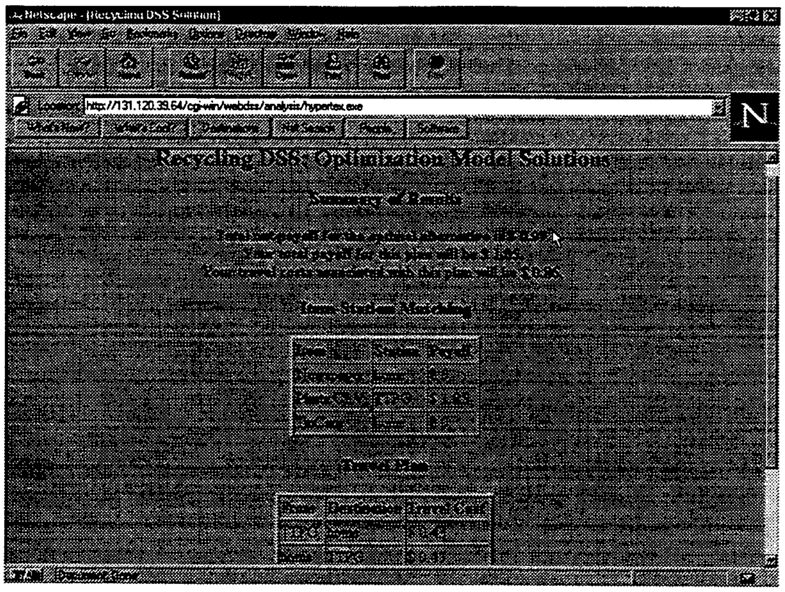
Recycling DSS Optimization

To see the AMPL generated output of the results, go to the [illegible] page. This program can take up to 10 minutes to compute the optimal result due to the number of variables generated and possible alternatives. Be patient or [illegible] after a few minutes to see if your output file is complete. If you would like to convert your results to hypertext, go to the [illegible] page.

This application was created by [illegible] for Professor [illegible]

Generated on [illegible]

Figure 15. Recycling DSS Optimization Alternatives



http://131.120.39.64/cgi-win/webdss/analysis/hypertex.exe

Recycling DSS: Optimization Model Solutions

Summary of Results

Total net payoff for the optimal alternative is $[illegible]

The total payoff for this plan will be $ [illegible]

Your travel costs associated with this plan will be $0.86

Item-Station Matching

Item	Station	Payoff
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	$ [illegible]
[illegible]	[illegible]	[illegible]

Travel Plan

From	Destination	Travel Cost
[illegible]	[illegible]	$ [illegible]
[illegible]	[illegible]	$ 0.[illegible]

Figure 16. Recycling DSS Solution

C. CGI SCRIPTS IN THE RECYCLING DSS

This section describes the CGI scripts necessary to facilitate user interaction with the Recycling DSS through the World Wide Web. These scripts are executable programs that communicate with Web documents and extend their functionality to produce dynamic and interactive pages.

The CGI scripts are divided by the functional area of the Recycling DSS in which they are incorporated. This follows the structure of the DSS and provides a logical division of functions. There are no executable functions utilized in the Education area of the Recycling DSS, so only the scripts performed in the Information and Analysis areas will be described.

All of the scripts, since they all are designed to operate via CGI, have the same basic structure. Each scripts has an HTML header section, a body, and a footer section. The HTML header section contains error-handling instructions, some of which are specific to the type of server software being used, as well as some statements that identify the program as being CGI-capable. The body section contains instructions on receiving and handling data, performing calculations, and producing dynamic HTML pages. In order to pass information from one program to another without requiring the user to reenter data, hidden fields are embedded into the forms that pass data to CGI scripts. The footer section closes out the application and sends a statement of Web document authorship.

1. Queries Section Scripts

a. List of Services Script

The list of services script (svcinfo.exe) receives no input data. Selection of this function on the control panel returns a listing of all of the available recycling services in the local area. The information is provided to the user in a table format utilizing a function of the CGIDB component which retrieves a table specified in the code, converts it to HTML and sends it back to the user in a format viewable on the WWW. The function of the script is to:

- Send the table of services to a dynamic HTML page, displaying the service name, city, address, type of service, and phone number of every available service in the local area.

b. Service Information Query

The service query script (svcquery.exe) is used to provide users with service information on a specific recycling service. The user selects the name of the recycling service from a list of options in a drop-down list. The user input is utilized as a basis for querying the database and returning the associated output. The script proceeds as follows:

- Retrieve the user input (aService) from the form.
- Construct an SQL statement on the fly using the user's input as the basis of the query.
- Send the query results for an HTML page in the form of a table.

c. Item Information Query

The item information query script (itmquery.exe) is used to provide users with information on the services with the capability to recycle a specific item. The user selects the type of recyclable material from a list of options in a drop-down list. The user input is utilized as a basis for querying the appropriate database and returning the associated output. The program performs the following:

- Retrieve the user input (theItem) from the form.
- Construct an SQL statement on the fly using the user's input as the basis of the query.
- Send the query results for an HTML page in the form of a table.

d. Distance Information Query

The distance information query script (distquer.exe) is used to provide users with distance information in miles to every available recycling service. The user's location is retrieved through the use of an image map in which the user places a cursor over their location on a geographic map of the area and clicks the left mouse button. The coordinates are used to calculate the distances to the recycling services through a conversion routine and the results are displayed to the user in a table format.

- Retrieve the pixel coordinates from the image map as a query string.
- Loop through the query string to determine the coordinate values by testing for the comma in the string and rebuilding the integer values into X and Y coordinates.
- Generate a query on the fly of the fields required of the appropriate table.
- Retrieve each record of the query result and assign them to values of an array of records.
- Utilize the pixel coordinate values for each record to calculate the distance from the pixel coordinates of the user using the Pathagorem theorem. Convert the results from pixels to miles.
- Produce an HTML table with the results by indexing each record of the array.

2. Analysis Section Scripts

a. Map Coordinates Script

The map coordinates script (recycle5.exe) provides a function similar to the distance information query script. The location of the consumer are grabbed from an image map and converted into a form usable by the DSS. The coordinates are required because they are used to determine distances to the recycling services. Due to the inherent nature of an image map, no other information can be obtained from the user. The requires the script to generate an HTML form to obtain more required data from the user. The script proceeds as follows:

- Receive pixel coordinates as a CGI Query string from an image map.
- Convert CGI Query string to separate X and Y pixel coordinates.
- Generate HTML form for further consumer input.
- Pass X and Y coordinates as hidden input to next script.

b. *Item Inputs Script*

This script (rpdsinpt.exe) retrieve the information inputted by the consumer on the form and utilizes the information to generate the appearance of the form it produces. The script uses the consumer's input for the number of items to recycle to generate a form to allow the consumer to provide types and amounts of materials to recycle. The script proceeds as follows:

- Receive input information from previous form.
- Generate a form, dependent on value of number of items to recycle, to allow the consumer to input information on the types and amounts of materials to reycle.
- Pass input information to next executable program as hidden data.

c. *Analysis Options Script*

The purpose of this script (options.exe) is to provide the user with a descriptive list of available anaylsis features and a means to execute the various options. Each analysis feature is provided to the user with a brief description of the type of solution the program will calculate. The functions of the script are as follows:

- Retrieve all inputted information from previous scripts.
- Generate a table of analysis options with associated form elements for user execution. Link appropriate hidden data with each form in order to pass required fields to executable script for processing.

d. Lazy Heuristic Script

The Lazy Heuristic script (lazyh.exe) determines solutions for the problem which provide visitation of a single recycling station to dispose of each selected item. The program does this by performing selective queries of the database tables which retrieve only records of recycling stations with payoff rates greater than or equal to zero for the items selected. Once these records are retrieved, basic calculations to determine payoffs and travel distances are performed and the results are returned to the user. The script accomplishes this by performing the following:

- Retrieve all pertinent hidden data passed from previous form.
- Generate queries of payoff rate and distance tables based on the items selected by the user.
- Calculate payoffs for each item at each station and the travel distance from the users' location to the appropriate station.
- Calculate utility for visitation of each recycling station.
- Generate table of results ordered by utility value.

e. Payoff Matrix Script

This script (rpdsout2.exe) calculates payoff and distance information for each recycling station and recyclable item and formats the output into a payoff matrix table. This provides the user with all payoff data for stations that are able to recycle at least one item selected. The program does this by using the hidden users input to generate several queries of the database for relevant data. The data generated is used to calculate item payoffs and home-to-station distances. These payoff values come from a table of payoff rates which is multiplied by the amounts entered for each item. The distance information is determined by the pixel coordinates passed along compared with the coordinates of the various services available. The program performs the following:

- Retrieve all hidden data passed along from previous forms.

- Retrieve all recycling item information and put information into two separate arrays.
- Perform query of databases with a table join.
- Calculate the payoffs for each item at each service.
- Calculate the distance to each recycling service from users' present location.
- Query database for curbside disposal of material dependent on responses inputted by user.
- Calculate payoffs for each item if curbside service is available.
- Calculate total payoff by summing item payoff values.
- Order records by total payoff value.
- Generate HTML output of solutions in a table format.

f. Optimization Script

The optimization program (datafile.exe) is the most complex script in the Recycling Decision Support System. This program takes the information passed from the previous executable to generate the data in a format compatible with the mathematical programming language. The modeling language (AMPL) requires the input of a model file and a data file in a specific format for proper execution. The script creates a data file in the proper format with a filename inputted by the consumer. In order to execute AMPL, a batch file is created with all of the commands and locations of the model and data files for proper execution. AMPL is commanded to execute optimization utilizing this batch file. The solution to the optimization program is formatted and written to an output file as specified by the batch file commands. The script proceeds as follows:

- Retrieve the data passed along by previous executable program.
- Perform query on database table and calculate payoffs for each service and item combination. Place values in records of an array.
- Perform query on database table dependent on input from consumer and calculate the payoffs as prescibed previously.

- Create file with data previously calculated. Perform third query to determine distance information and append this information to the data file.
- Close data file and create a batch file including commands to include model and data files, specify solver, generate output file, and format output.
- Execute AMPL through a command line.
- Generate HTML to specify location of AMPL generated output file.
- Generate form to allow user to process the output file in hypertext.

g. Hypertext Script

The hypertext script (hypertex.exe) translates the AMPL generated output file into a hypertext document with appropriate tags to provide clients solutions in an understandable format. The Hypertext script performs the following:

- Retrieve the file name and path for the AMPL generated output file.
- Open the output file and capture the file line by line as strings.
- Parse data from the file into array elements.
- Encapsulate data with hypertext tags and return HTML to client 'on the fly'.

3. Provider Section Scripts

a. Provider Information Scripts

The Provider section incorporates two scripts (infoupdt.exe and rateupdt.exe) which provide idenitical functions except on separate pieces of data from the database. These scripts serve the purpose of allowing providers (recycling stations) to update their administrative and payoff rate information real-time from a remote location. Each script retrieves and verifies the providers' identication and password information from the previous form input fields. Using this information, a query is generated on either the administrative information or payoff rate database tables which retrieves the providers' specific record. The program regenerates a form with all of pertinent fields available for alteration by the provider. The scripts flow as follows:

- Retrieve provider ID and password from previous form.
- Verify ID and password with database information. If information does not match, send entry error response to user.
- If data is correct, generate query on appropriate database table using providers' ID as the key.
- Generate HTML form of using fields of retrieved record as input variables.

b. Provider Update Scripts

The update scripts (update.exe and update2.exe) serve the purpose of retrieving the providers' entered information from the previous script and modifying the appropriate database table information with the updated data. Each script performs the same function, but on different database tables. The scripts perform the following:

- Retrieve form input information from previous script.
- Generate a query of the appropriate database table and update the associated record with the new data.

VI. CONCLUSION

The emergence of the World Wide Web as the primary mechanism for global connectivity and communications offers exciting opportunities in the areas of education and business alike. Unfortunately, utilization of the WWW beyond printed propaganda, downloadable files and images, is well below its potential. Currently, most Web servers are about as interactive as their paper-based counterparts. In order for the Web to become the next-generation platform for distributed computing, it must go beyond read-only into the world of interactive [Ref. 22].

For the individual or organization wishing to employ a scientific approach in solving decision problems, there is a plethora of relevant concepts, methods, models, and software [Ref. 5]. Yet, decision technologies are little used in real-world decision making. This may be on the verge of changing as decision support is taking on new importance within the IT strategy of most organizations. The demand for better information is fueled by fundamental changes in the business environment [Ref. 25]. Soon, the Internet and corporate intranets will become the primary vehicles to provide enterprise wide decision support, transitioning from proprietary client-server schemes to client-server implementations based on Web technology [Ref. 26].

This thesis attempts to resolve some of these issues by exploring the issues and processes involved in the development of a Web-based decision support system. Utilizing rapid application development tools, a prototype DSS was created and implemented over the WWW to support ordinary people and industrial users in making good decisions for recycling and disposal of household and industrial waste. This prototype system is currently functional and on line at *http://dnet2.sm.nps.navy.mil/-webdss.* In examining the Recycling DSS, new framework and architectures for Web-based decision support systems are revealed.

A. CONTRIBUTIONS

The prototype development and modification process utilized to construct the Recycling DSS resulted in a fully functional, real-world application which provides tremendous benefits to the population of the local Monterey Bay area. Generalization of the models, architecture, and data sets utilized in this specific system enable the capability for other communities and geographic regions to adopt this system for localized implementation.

Analysis of the system highlights a generalized framework for Web-based decision support emphasizing the need to address the functional areas of education, queries and analysis. It is not enough for a Web-based DSS to provide mathematical analysis of specific decision problems, the system must facilitate learning and individual cognitive processing. The areas of education and queries should sufficiently address these processes for a system to be complete.

Technical evaluation of the system uncovered a new architecture for Web-based decision support systems. In contrast to the standard DSS, a Web-based DSS architecture utilizes functional agents to carry out the processes of communication, translation and data transfer between functional components. The additional language translation requirements of HTML and protocol standards of HTTP for Web connectivity, necessitate the inclusion of these agents.

B. AREAS OF ADDITIONAL RESEARCH

This thesis has brought up several areas of additional research. In concert with the technological revolution, no technology is stagnant. As the developments tools, computational processing capabilities and software applications become more advanced, decision support systems should follow. A few of these areas which take advantage of current technology are listed below.

- Generalized DSS: The success of the prototype system developed lends credence to the issue of development of a generalized Web-based decision support system. Such a system would seemingly become invaluable to corporate and academic societies.

- Java technology: The explosion of Java as the new trend in Web computing offers great possibilities for Web-based DSS support. Providing the benefits platform independence and Web connectivity, development of a DSS application in Java may solve some of the problems of server-side computing.

- API/CORBA: Although CGI holds a commanding position as leading interface agent, API and CORBA are gaining ground as alternative interface mechanisms which require less overhead and are not as computationally intensive. Development of Web-based decision support systems utilizing these interfaces provide an interesting study in relative performance.

C. CONCLUSION

The World Wide Web will continue to expand in both nodes and users. As the Web grows, the potential uses for Web connectivity such as supporting decision technologies will be realized. Web-based decision support systems overcome many of the problems rooted in conventional desktop decision support systems and create new opportunities for the increased use of scientific methods in decision-making. Enabling decision support systems over the WWW or over organizational Intranets using the same technologies as the WWW, may be the key to establishing a permanent and indispensable place for the DSS in todays organizational information system foundation.

LIST OF REFERENCES

[1] T. Berners-Lee, R. Cailliau, A. Luotonen, N. Frystyk, and A. Secret. The world wide web. *Communications of the ACM*, 37(8):76–82, 1994.

[2] H.K. Bhargava, R. Krishnan, and R. Müller. On parameterized transaction models for agents in electronic markets for decision technologies. In Sudha Ram and M. Jarke, editors, *Proceedings of the Fifth Workshop on Information Technologies and Systems, Amsterdam, Holland, December 1995*, pages 218–227. RWTH Aachen, Fachgruppe Informatik, 1995.

[3] E. Krol and P. Fagun. *The Whole Internet for Windows 95.* O'Reilly & Associates, 1995.

[4] R.H. Sprague and E.D. Carlson. *Building Effective Decision Support Systems.* Prentice Hall, 1982.

[5] H.K. Bhargava, R. Krishnan, and R. Müller. Decision support on demand: Emerging electronic markets for decision technologies. *Decision Support Systems*, forthcoming:xx, 1997.

[6] A.S. King. Decisionnet–a prototype distributed decision support system server. Master's thesis, Naval Postgraduate School, 1995.

[7] Webmaster@consrv.Ca.Gov. Recycling information, 1996. http://www.consrv.-ca.gov/dor/recycle.html.

[8] Mureg3@uxa.Ecn.Bgu.Edu. Recycling should be influenced by government policies, 1996. http://www.ecn.bgu.edu/mureg3/about-me/recycling.htm.

[9] R. Krishnan H.K. Bhargava and A. Whinston. On integrating collaboration and decision analysis techniques. *Journal of Organizational Computing*, 4(3):297–316, 1994.

[10] S.E. Bodily. *Modern Decision Making: A Guide to Modeling with Decision Support Systems.* McGraw-Hill, 1985.

[11] R. Fourer, D.M. Gay, and B.W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming.* Boyd & Fraser, 1993.

[12] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization.* John Wiley & Sons, 1985.

[13] G. Ariav and M.J. Ginzberg. Dss design: A systemic view of decision support. *Communications of the ACM*, 28(10):1045–1052, 1985.

[14] H. Berghel. The clients side of the world wide web. *Communicaitons of the ACM*, 39(1):31–40, 1996.

[15] C. Liu, J. Peek, R. Jones, B. Buus, and A. Nye. *Managing Internet Information Services.* O'Reilly & Associates, 1994.

[16] Httpd@ncsa.Uiuc.Edu. Ncsa imagemap tutorial, 1996. http://hoohoo.-ncsa.uiuc.edu/docs/tutorials/imagemapping.html.

[17] M.S. Morris and R.J. Hindrichs. *Web Page Design.* Sun Microsystems, 1996.

[18] S.H. Earley. Decisionnet: A database approach. Master's thesis, Naval Postgraduate School, 1996.

[19] M. Bieber and C. Kacmar. Designing hypertext support for computational applications. *Communications of the ACM*, 38(8):99–107, 1995.

[20] R. McDaniel. Cgi manual of style, 1996. http://www.mcp.com/zdpress/-features/3970/Content/Html/ch01.htm.

[21] J.T. Casler. Infrastructure considerations for world wide web servers. Master's thesis, Naval Postgraduate School, 1996.

[22] E. Hall. Adding interactive services to your web server. *Network Computing*, (701):66, January 15 1996.

[23] X. Pacheco and S. Teixeira. *Delphi Developers Guide.* Sams Publishing, 1995.

[24] A. Lynnworth. Delphi cgi component. *http://super.sonic.net/ann/delphi-/cgicomp/detail.htm*, 1995.

[25] E. Woods. Text of ovum opap white paper. *Techwire*, page 12, January 22 1997.

[26] E. Kay. Business analysis–decisions, decisions–more business analyst use their query and reporting tools to more efficiently use data warehouses. next up: Internet query tools. *Communications Week*, (584):66, June 17 1996.

INITIAL DISTRIBUTION LIST

1.	Defense Technical Information Center 8725 John J. Kingman Rd., Ste 0944 Ft. Belvoir, VA 22060-6218	2
2.	Dudley Knox Library Naval Postgraduate School 411 Dyer Rd. Monterey, CA 93943-5101	2
3.	Professor Hemant Bhargava (Code SM/BH) Systems Management Departement Naval Postgraduate School Monterey, CA 93940-5000	3
4.	Professor Suresh Sridhar (Code SM/SR) Systems Management Department Naval Postgraduate School Monterey, CA 93940-5000	1
5.	Professor Ramayya Krishnan The Heinz School Carnegie Mellon University Pittsburgh, PA 15213	1
6.	Professor Rudolf Müller Institüt für Wirtschaftsinformatik Humboldt-Universität zu Berlin Spandauer Str. 1 10178 Berlin, Germany	1
7.	Professor Gordon H. Bradley (Code OR/BZ) Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5000	1
8.	Professor Arhtur M. Geoffrion Decision Sciences John E. Anderson Graduate School of Management Box 951481, UCLA Los Angeles, CA 90095-1481	1

9. Professor Kevin Gue (Code SM/GK) Department of Systems Management Naval Postgraduate School Monterey, CA 93940-5000	1
10. Professor Andrew B. Whinston Graduate School of Business Center for Information Systems Management University of Texas at Austin Austin, TX 78713-8916	1
11. LCDR Clay G. Tettelbach 8260 Toll House Rd. Annandale, VA 22003	1